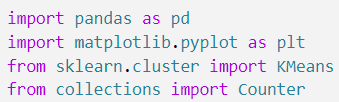
Ερώτημα 2

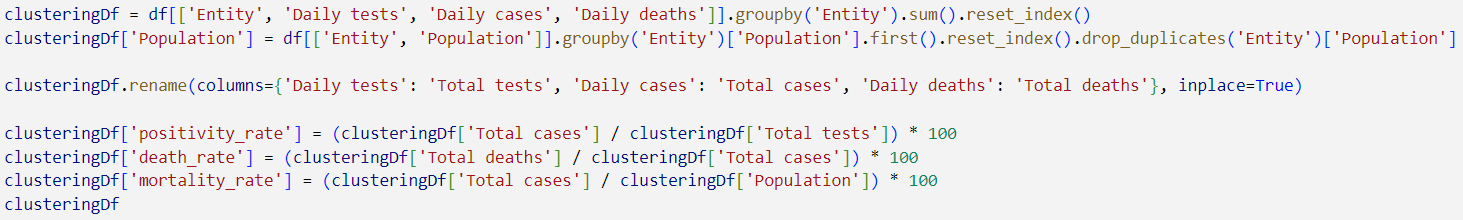
Αρχικά κάνουμε import τις κατάλληλες βιβλιοθήκες και κλάσεις που θα χρειαστούμε.



Την pandas για την επεξεργασία των .csv αρχείων ως dataframes, την matplotlib.pyplot για τη σχεδίαση των γραφικών παραστάσεων, την κλάση KMeans από τη βιβλιοθήκη sklearn.cluster για την συσταδοποίηση των δεδομένων μας και τέλος την κλάση Counter από τη βιβλιοθήκη collections για τη καταγραφή του αριθμού των entities που περιέχονται σε κάθε cluster.



Στη συνέχεια φορτώνουμε το dataset αφού έχει περάσει από την προ-επεξεργασία στο προηγούμενο ερώτημα.



Έπειτα δημιουργούμε ένα clustering dataframe όπου επιλέγουμε τις στήλες Entity, Daily tests, Daily cases και Daily deaths και κάνουμε sum aggregation με βάση το Entity και τέλος reset index.

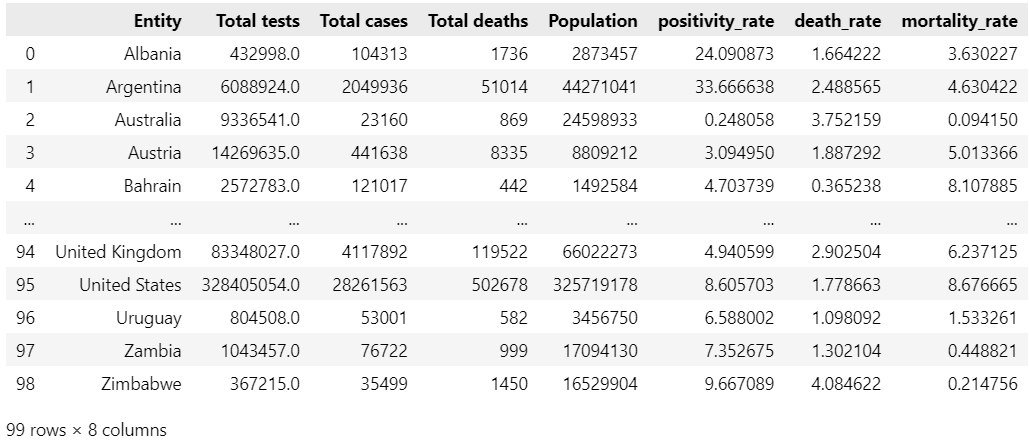
Δημιουργούμε μια νέα στήλη για το Population όπου παίρνουμε τον πληθυσμό για κάθε Entity και ακολουθούν οι κατάλληλες εντολές reset index και drop duplicates για να μετατραπεί το dataframe σε μία στήλη. Συνεχίζοντας αλλάζουμε το όνομα των στηλών και υπολογίζουμε τα απαραίτητα μεγέθη που ζητάει η εκφώνηση.

Το ποσοστό θετικότητας (positivity rate) ισούται με το πηλίκο των συνολικών cases διά των συνολικών tests επί 100%

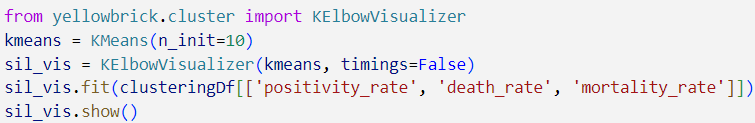
Το ποσοστό θνησιμότητας (death rate) ισούται με το πηλίκο των συνολικών θανάτων διά των συνολικών cases επί 100%

Ο αριθμός κρουσμάτων σε σχέση με τον πληθυσμό της χώρας (mortality rate) ισούται με το πηλίκο των συνολικών cases διά του πληθυσμού κάθε χώρας επί 100%

Το dataframe που προκύπτει είναι το εξής:

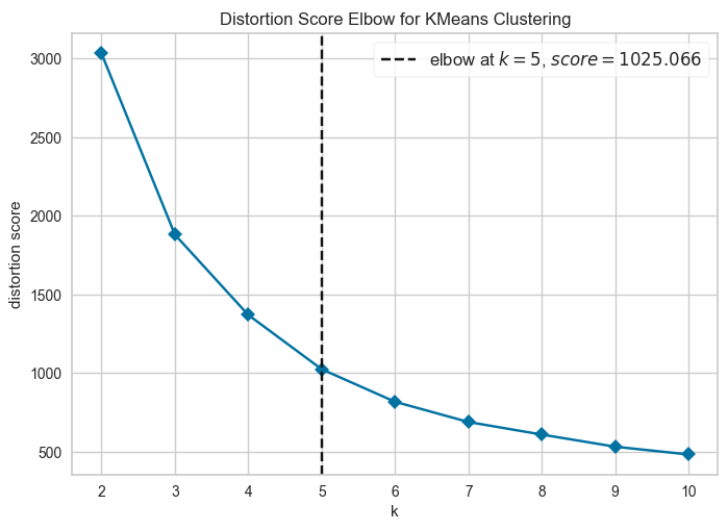


Ο αλγόριθμος συσταδοποίησης που χρησιμοποιήθηκε είναι ο KMeans καθώς τα δεδομένα εισόδου (positivity rate, death rate και mortality rate) είναι ευκλείδια. Δεν χρησιμοποιήθηκε το longitude και latitude καθώς είναι γνωστό ότι ακολουθούν σφαιρική γεωμετρία που είναι ένα παράδειγμα μη ευκλείδιας γεωμετρίας και ο KMeans δέχεται μόνο ευκλείδια δεδομένα.



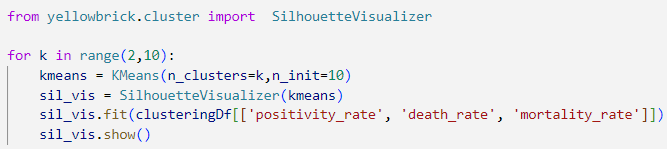
Ο KElbowVisualizer υλοποιεί την elbow μέθοδο για να μας βοηθήσει να επιλέξουμε το βέλτιστο αριθμό συστάδων προσαρμόζοντας το μοντέλο με ένα διάστημα τιμών για το K. Αν το διάγραμμα μοιάζει με ένα χέρι, τότε ο «αγκώνας» (το σημείο κάμψης της καμπύλης) είναι μια καλή ένδειξη ότι το υποκείμενο μοντέλο προσαρμόζεται καλύτερα σε αυτό το σημείο. Στο παρακάτω γράφημα ο «αγκώνας» θα σημειωθεί με μια διακεκομμένη γραμμή.

Από προεπιλογή, η μετρική της παραμέτρου βαθμολόγησης έχει οριστεί ως το distortion (παραμόρφωση), η οποία υπολογίζει το άθροισμα των τετραγωνικών αποστάσεων από κάθε σημείο προς το κέντρο που του έχει ανατεθεί.



Από το γράφημα βλέπουμε ότι ο KElbowVisualizer μας υπέδειξε ότι ο βέλτιστος αριθμός συστάδων είναι 5.

Ας εξετάσουμε και μια άλλη μετρική που ονομάζεται Silhouette Coefficient.



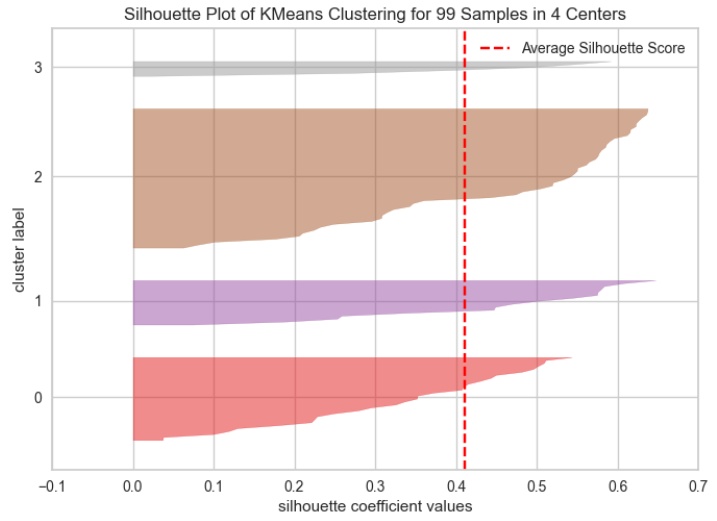
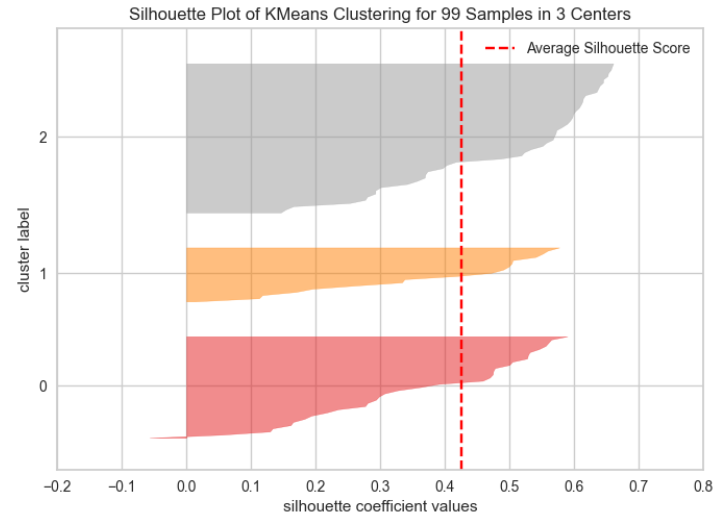
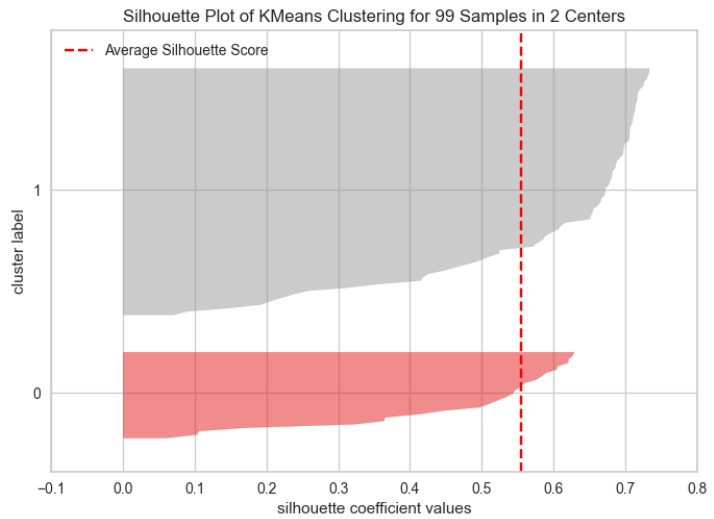
Χρησιμοποιούμε τώρα τον SilhouetteVisualizer και θέλουμε να παρατηρήσουμε από τα γραφήματα που θα προκύψουν για k από 2 έως 10 ποιος αριθμός συστάδων είναι ο βέλτιστος.

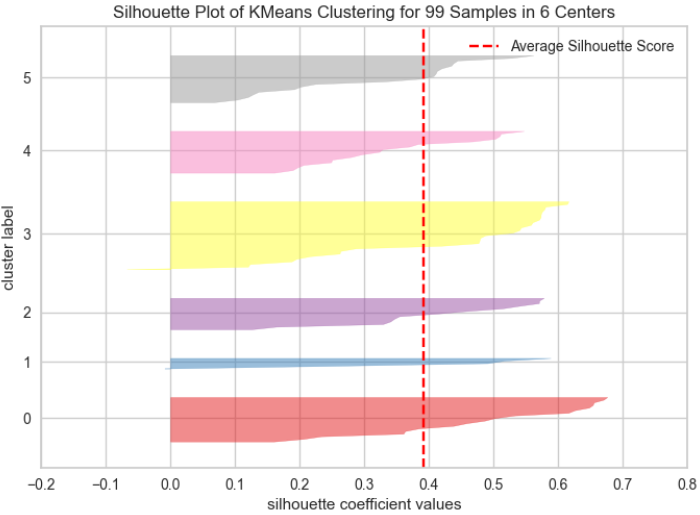
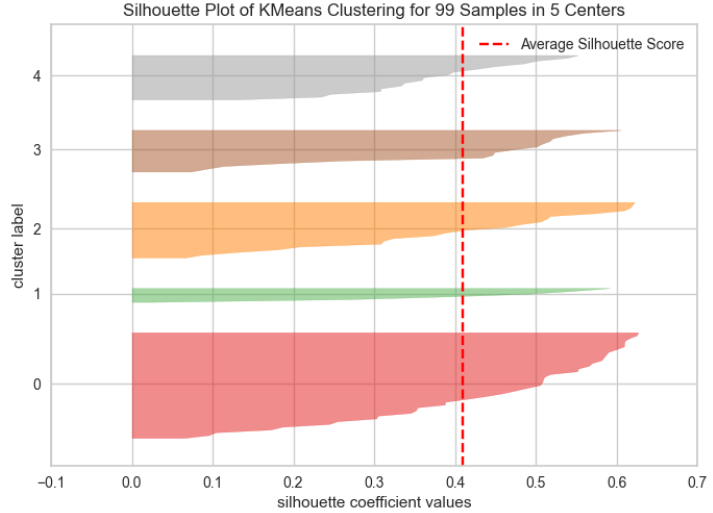
Ο SilhouetteVisualizer εμφανίζει το silhouette coefficient για κάθε δείγμα ανά συστάδα αξιολογώντας οπτικά την πυκνότητα και τον διαχωρισμό μεταξύ των συστάδων. Η βαθμολογία υπολογίζεται με τη μέση τιμή του silhouette coefficient για κάθε δείγμα, ο οποίος υπολογίζεται ως η διαφορά μεταξύ της μέσης απόστασης εντός συστάδα και της μέσης απόστασης πλησιέστερης συστάδας για κάθε δείγμα, κανονικοποιημένη από τη μέγιστη τιμή. Αυτό παράγει μια βαθμολογία μεταξύ -1 και +1, όπου βαθμολογίες κοντά στο +1 υποδηλώνουν υψηλό διαχωρισμό και βαθμολογίες κοντά στο -1 υποδηλώνουν ότι τα δείγματα μπορεί να έχουν ανατεθεί σε λάθος συστάδα.

Στα γραφήματα του SilhouetteVisualizer οι συστάδες με υψηλότερες βαθμολογίες έχουν πιο wide silhouettes, αλλά οι συστάδες που είναι λιγότερο συνεκτικές θα υπολείπονται της μέσης βαθμολογίας από όλες τις συστάδες, η οποία απεικονίζεται ως μια κατακόρυφη κόκκινη γραμμή.

Άρα για να διαλέξουμε το κατάλληλο K θέλουμε τα clusters να απεικονίζονται ομοιόμορφα wide, να είναι πάνω από την κόκκινη διακεκομμένη γραμμή και να μην πηγαίνουν πίσω από το 0 που δηλώνει ότι έχει γίνει λάθος στο clustering και κάποια δείγματα κανονικά ανήκουν σε άλλη συστάδα.

Οι γραφικές που προκύπτουν από το παραπάνω κομμάτι κώδικα είναι:



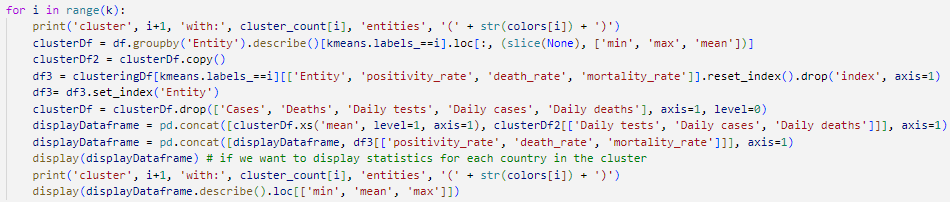


Παρατηρούμε ότι σε όλα τα διαγράμματα όλα τα clusters περνάνε την κόκκινη κατακόρυφη γραμμή. Στα διαγράμματα με K=3, 6 υπάρχουν clusters όπου έχει γίνει λάθος επειδή συνεχίζουν και πριν το 0 οπότε τα απορρίπτουμε. Στο διάγραμμα με K=2 υπάρχει μεγάλο imbalance αφού το ένα είναι πολύ πιο παχύ από το άλλο. Τα διαγράμματα με K=4 και K=5 είναι πολύ παρόμοια αλλά φαίνεται στο διάγραμμα με K=4 τα clusters να μην είναι τόσο ομοιόμορφα στο πάχος κι αφού και από την προηγούμενη μετρική με το KElbowVisualizer βρήκαμε ότι το βέλτιστο K=5 τότε αυτό θα επιλέξουμε τελικά.

Τέλος υλοποιούμε ένα 3d scatter plot για κάθε πιθανό K των σημείων κάθε συστάδας ώστε να δούμε καλύτερα τη διάταξη τους στο χώρο και να παρατηρήσουμε καλύτερα την ομαδοποίηση που έχει πραγματοποιηθεί.

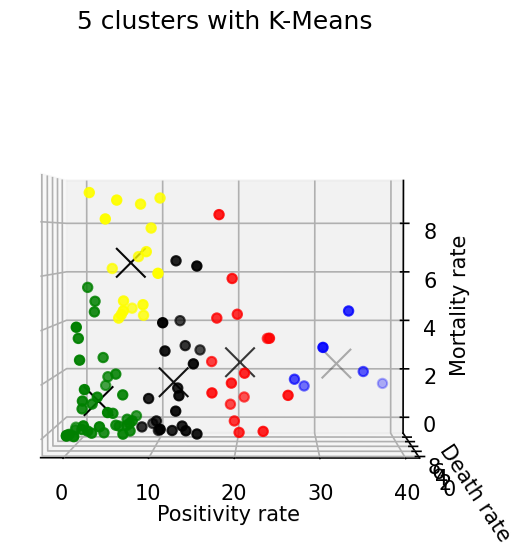
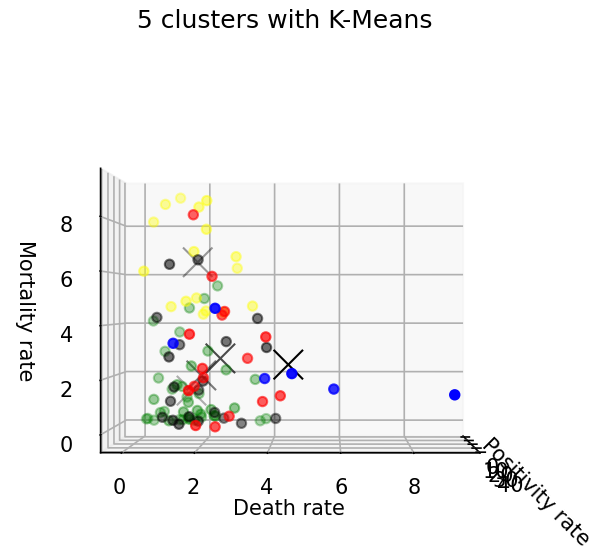


Κάνουμε πάλι clustering και φτιάχνουμε τα plots ορίζοντας κατάλληλα χρώματα για τα δείγματα κάθε συστάδας, τα απαραίτητα labels και το που βρίσκονται τα κεντροειδή.



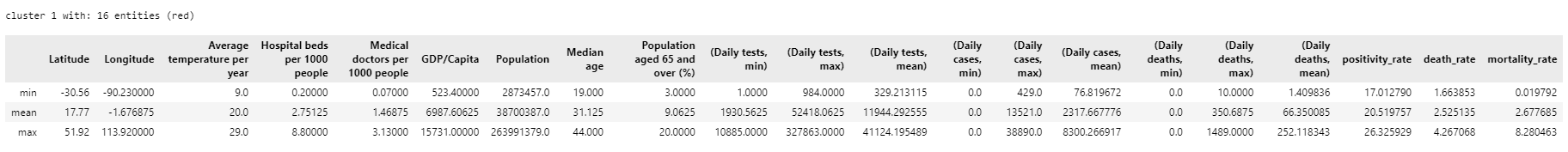
Για κάθε συστάδα βρίσκουμε τα απαραίτητα στατιστικά στοιχεία που χρειαζόμαστε και ποιες χώρες βρίσκονται σε αυτή.

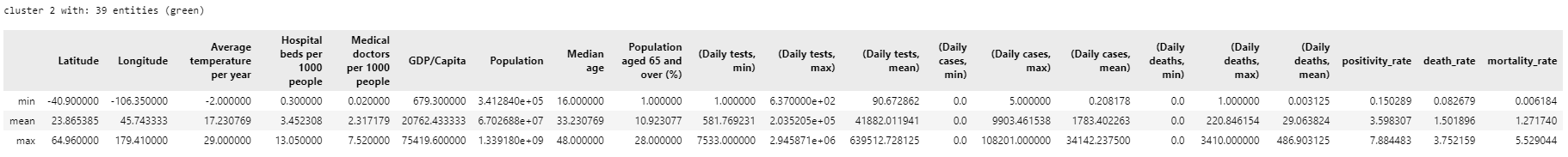
Εμείς θα επικεντρωθούμε στην γραφική για K=5 clusters που βρήκαμε ότι είναι το βέλτιστο

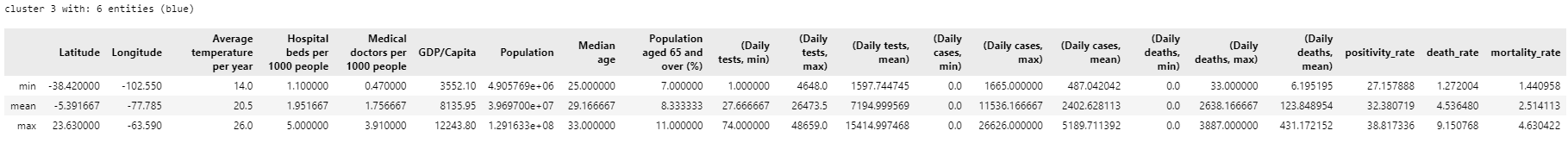
 

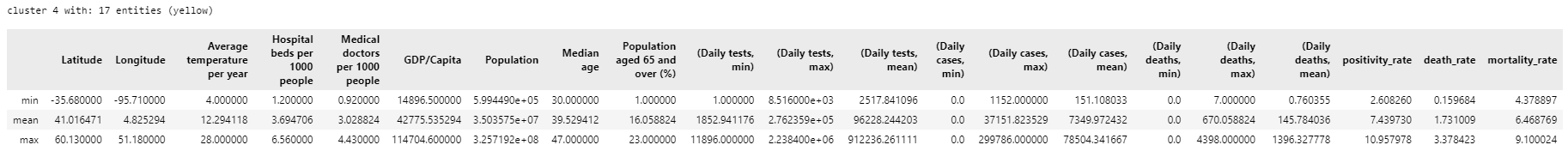
Από τα διαγράμματα φαίνεται ότι το μπλε cluster έχει υψηλές τιμές positivity rate και death rate ενώ το κίτρινο cluster φαίνεται ότι έχει υψηλές τιμές mortality rate.

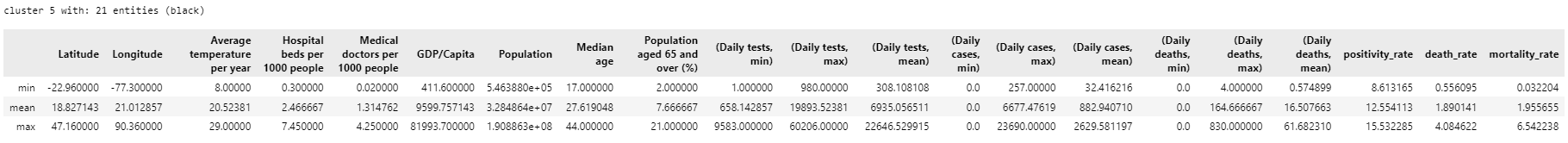
Οι στατιστικές τιμές για κάθε cluster παρατίθενται παρακάτω.





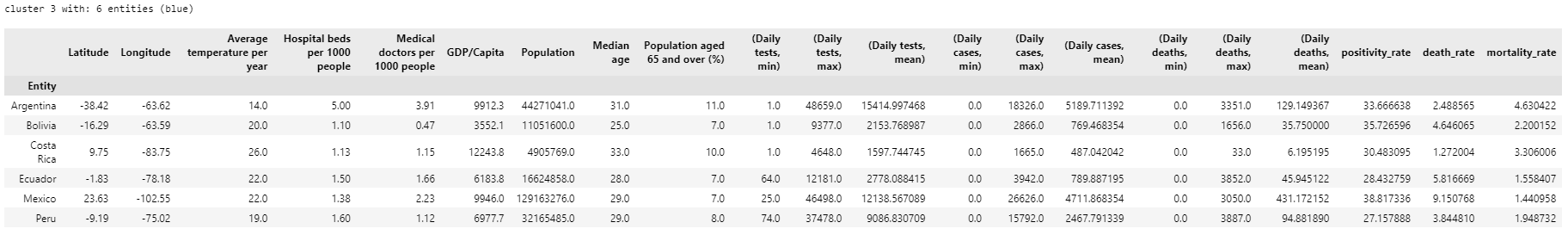






Θα απαντήσουμε πρώτα στην ερώτηση αν υπάρχουν χώρες που ξεχωρίζουν αρνητικά. Για να ξεχωρίσει μια συστάδα αρνητικά θα πρέπει να έχει υψηλές mean values και στα τρία πεδία (positivity rate, death rate και mortality rate).

Παρατηρούμε ότι η συστάδα 3 έχει τη μεγαλύτερη μέση τιμή για positivity rate και death rate από όλες τις άλλες συστάδες και την τρίτη μεγαλύτερη μέση τιμή για mortality rate άρα δίκαια μπορεί να χαρακτηριστεί ως η συστάδα που ξεχωρίζει αρνητικά. Άρα οι 6 χώρες που περιέχει ξεχωρίζουν αρνητικά και είναι οι εξής:

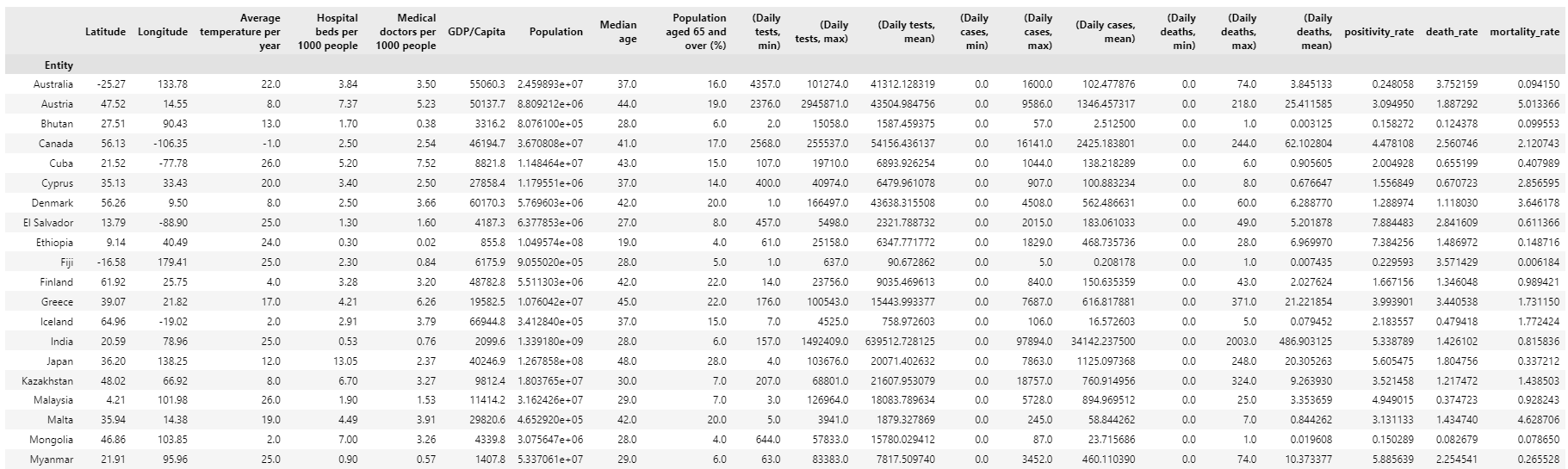
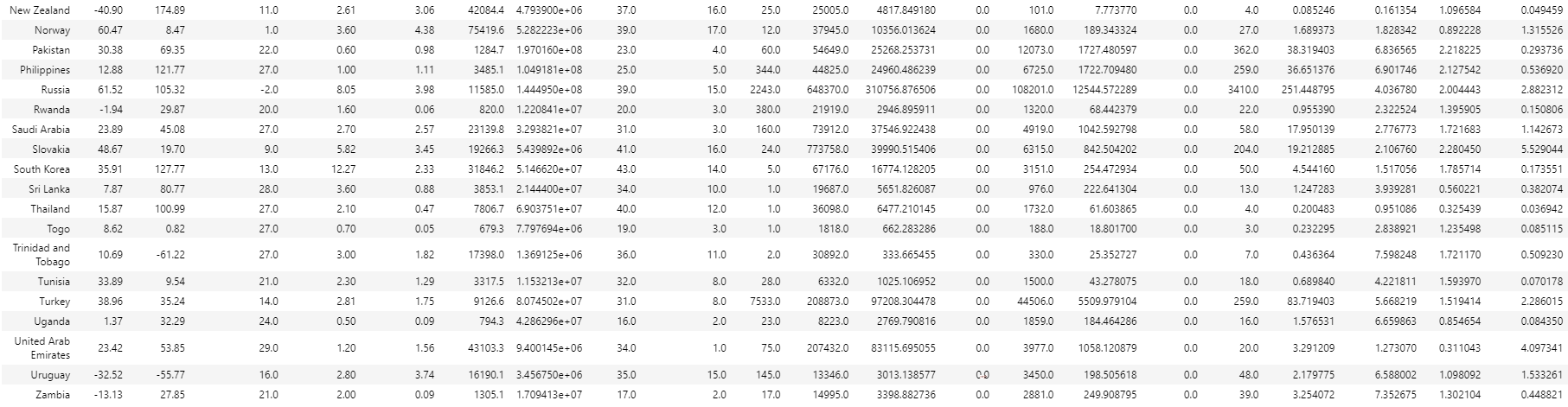


Τα κοινά χαρακτηριστικά που έχουν είναι ότι βρίσκονται όλες εκτός από το μεξικό στη νότια αμερική (το οποίο μεξικό βρίσκεται πολύ κοντά γεωγραφικά στην κόστα ρίκα). Έχουν παρόμοια θερμοκρασία και λίγο αριθμό κρεβατιών και γιατρών. Επιπλέον έχουν χαμηλό GDP/Capita.

Η χώρα με την χειρότερη κατάσταση που μπορεί να θεωρηθεί ως outlier είναι το Μεξικό καθώς συμβάλει στη μεγαλύτερη τιμή του positive και death rate για τη συστάδα του.

Τώρα θα απαντήσουμε στην ερώτηση αν υπάρχουν χώρες που ξεχωρίζουν θετικά. Για να ξεχωρίσει μια συστάδα θετικά θα πρέπει να έχει χαμηλές mean values και στα τρία πεδία (positivity rate, death rate και mortality rate).

Παρατηρούμε ότι η συστάδα 2 έχει τη μικρότερη μέση τιμή και για τα τρία πεδία άρα δίκαι μπορεί να χαρακτηριστεί ως η συστάδα που ξεχωρίζει θετικά. Άρα οι 39 χώρες που περιέχει ξεχωρίζουν θετικά και είναι οι εξής:

Επειδή είναι πολλές οι χώρες δεν μπορούμε να εντοπίσουμε κοινά χαρακτηριστικά μεταξύ τους. Αυτό που μπορούμε να κάνουμε είναι να συγκρίνουμε το μέσο όρο μερικών χαρακτηριστικών με τον αντίστοιχο μέσο όρο αυτών των χαρακτηριστικών για τη συστάδα 3.

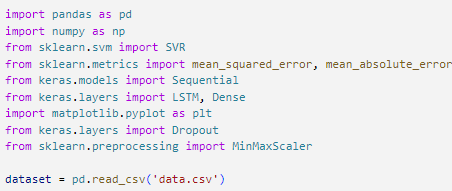
Από τις παραπάνω εικόνες για τα στατιστικά των δύο χωρών μπορεί να γίνει αντιληπτό ότι η ιατρική περίθαλψη στις χώρες της συστάδας 2 είναι καλύτερη από αυτή των χωρών της συστάδας 3 αφού κατά μέσο όρο έχουν περισσότερους γιατρούς και κρεβάτια. Επίσης κατά μέσο όρο το GDP/Capita είναι 2 φορές μεγαλύτερο της συστάδας 2 από τη συστάδα 3.

Ερώτημα 3

Για να πραγματοποιηθεί καθημερινή ανάλυση των δεδομένων θα πρέπει να έχει ήδη δημιουργηθεί ένας παλινδρομητής ώστε μετά να μπορεί να λαμβάνει τα δεδομένα που χρειάζονται κάθε μέρα και να προβλέπει το ποσοστό θετικότητας 3 μέρες μετά. Θεωρούμε ότι στην καθημερινή ανάλυση δεν μπορούν να ληφθούν υπόψιν ως είσοδος στο μοντέλο τα δεδομένα θετικότητας της ίδιας ημέρας, δηλαδή στην 1/1/2021 για να προβλέψουμε για της 4/1/2021 μπορούν να ληφθούν υπόψιν δεδομένα μέχρι και την 31/12/2020.

Ακολουθήθηκαν 2 προσεγγίσεις. Στη μία προσέγγιση ως δεδομένο εισόδου λήθφηκε υπόψιν μόνο η προηγούμενη ημέρα ενώ στη δεύτερη προσέγγιση ως δεδομένο εισόδου λήθφηκαν οι 3 προηγούμενες ημέρες.

Προσέγγιση 2

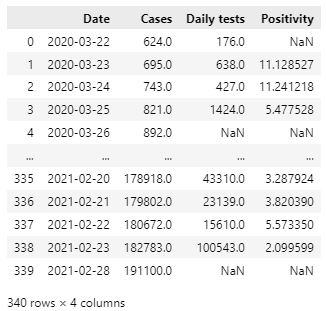


Αρχικά γίνονται import οι κατάλληλες βιβλιοθήκες για τη διαχείριση των δεδομένων (dataframes και numpy arrays), στη συνέχει βιβλιοθήκες που αφορούν τους παλινδρομητές (RNN και SVR), κάποιες μετρικές για την αξιολόγηση των παλινδρομητών και μία κλάση για την κανονικοποίηση των δεδομένων εισόδου. Στο τέλος φαίνεται το dataset που έχουμε σε αυτή την άσκηση και αφορά το αρχικό dataset αφού θέλουμε να έχουμε στη διάθεσή μας όλες τις ημερομηνίες σε συνεχόμενη σειρά για την Ελλάδα.



Στη συνέχεια από το dataset κρατάμε μόνο για την Ελλάδα τις στήλες Date, Cases, Daily tests και πετάμε μερικές τιμές από την αρχή και το τέλος ώστε να έχουμε καλύτερα αποτελέσματα στη συνέχεια. Επίσης μετατρέπουμε τη στήλη Date σε τύπο datetime και κάνουμε sort το dataframe σύμφωνα με αυτή τη στήλη. Τέλος δημιουργούμε μια καινούργια στήλη Positivity που κάθε της γραμμή ισούται με τη διαφορά των Cases στην ίδια και την προηγούμενη γραμμή διά τον αριθμό των daily tests στην ίδια γραμμή επί 100%.

Το dataset που προκύπτει είναι το εξής:

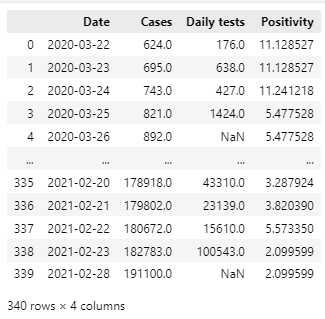


Για να αφαιρέσουμε τις NaN τιμές από την στήλη του Positivity χρησιμοποιούμε το παρακάτω κομμάτι κώδικα



Με αυτές τις εντολές κάθε NaN τιμή όταν χρησιμοποιείται το ffill (forward fill) παίρνει την τιμή της προηγούμενης γραμμής στο Positivity column. Το bfill (backwards fill) χρησιμοποιείται για να πάρει τιμή η NaN τιμή της γραμμής 0 καθώς δεν έχει προηγούμενη τιμή και παίρνει την τιμή της επόμενης γραμμής (11.128527).

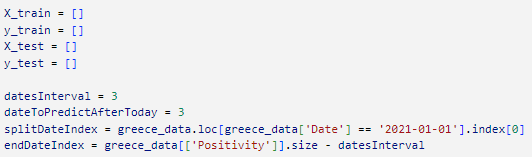
Το αποτέλεσμα που προκύπτει είναι το εξής:



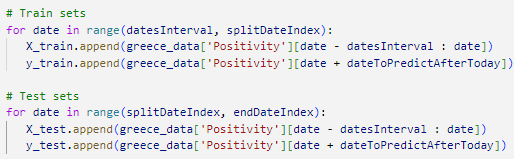
Τις NaN τιμές στην στήλη Daily tests δεν μας ενδιαφέρει να τις αλλάξουμε αφού η είσοδος στους παλινδρομητές θα είναι μόνο από την στήλη Positivity.



Δημιουργούμε έναν scaler ο οποίος κάνει map όλες τις τιμές της στήλης Positivity μεταξύ [0, 1]. Αυτό γιατί οι παλινδρομητές μας δουλεύουν καλύτερα με δεδομένα σε αυτό το διάστημα. Στη συνέχεια χρησιμοποιούμε το αντικείμενο scaler για να κάνουμε τον αντίστροφο μετασχηματισμό όταν θα θέλουμε να κάνουμε plot τα αποτελέσματα.

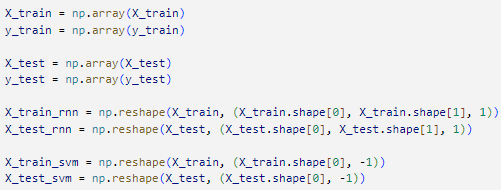


Με τις εντολές αυτές αρχικοποιούμε τα train και test sets, ορίζουμε ότι το datesInterval ισούται με 3, δηλαδή ότι θα λάβουμε 3 μέρες υπόψιν για την πρόβλεψή μας, και ότι το dateToPredictAfterToday ισούται με 3, δηλαδή ότι θέλουμε να κάνουμε πρόβλεψη για 3 μέρες μετά την κάθε μέρα της ανάλυσης. Το splitDateIndex μας δίνει τη γραμμή όπου υπάρχει η ημερομηνία 01/01/2021 που θέλουμε να ξεκινήσουμε την καθημερινή ανάλυση ώστε να χωρίσουμε τα δεδομένα σε train και test. Το endDateIndex αναλόγως το datesInterval ισούται με την τελευταία μέρα για την οποία μπορούμε να κάνουμε κάποια ανάλυση.



Σε αυτό το κομμάτι του κώδικα ορίζουμε τα train και test sets. Για X\_train και X\_test θα ληφθούν υπόψιν για κάθε μέρα που γίνεται ανάλυση οι 3 προηγούμενες με τη διαφορά ότι τα δεδομένα για το X\_test ξεκινάνε από 1/1/2021 και μετά. Για y\_train και y\_test θα ληφθεί υπόψιν για κάθε μέρα που γίνεται ανάλυση η τιμή του Positivity 3 μέρες μετά.

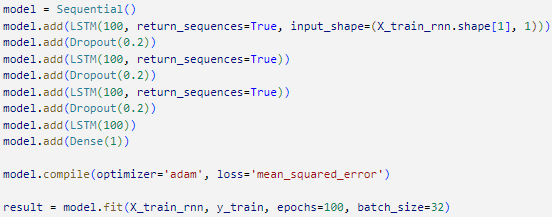
Άρα για είσοδο έχουμε κάθε φορά 3 μέρες positivity πριν την ημέρα της ανάλυσης και ως έξοδο έχουμε το positivity 3 μέρες μετά την ημέρα της ανάλυσης.



Με αυτό το κομμάτι κώδικα μετατρέπουμε τις λίστες σε numpy arrays αφού είναι πολύ πιο αποδοτικά από τις λίστες τις python για πράξεις και είναι ο μόνος αποδεκτός τύπος για τους παλινδρομητές.

Ύστερα για το RNN πρέπει να κάνουμε ένα reshape των δεδομένων εισόδου του training και του testing καθώς πρέπει να είναι σε κατάλληλη μορφή για να δοθούν στο νευρωνικό δίκτυο. Η πρόσθετη διάσταση μεγέθους 1 στο τέλος αντιπροσωπεύει έναν πρόσθετο άξονα, ο οποίος είναι χρήσιμος για την εισαγωγή ακολουθιών σε ένα επαναλαμβανόμενο νευρωνικό δίκτυο.

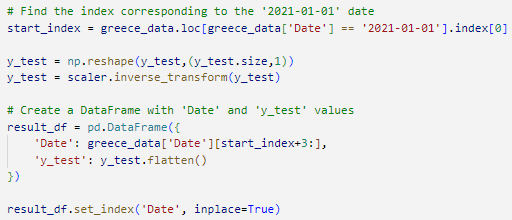
Για την είσοδο στον παλινδρομητή SVM ακολουθούμε την ίδια διαδικασία μόνο που το -1 σε αυτή την περίπτωση χρησιμοποιείται ως μία από τις διαστάσεις κι επιτρέπει στη NumPy να υπολογίζει αυτόματα το μέγεθος αυτής της διάστασης βάσει των άλλων διαστάσεων και του συνολικού αριθμού των στοιχείων του πίνακα.



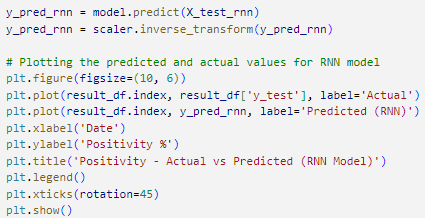
Για RNN νευρωνικό δίκτυο χρησιμοποιήσαμε το LSTM το οποίο βοηθάει στο πρόβλημα των παραδοσιακών RNNs που έχουν πολύ μικρή μνήμη και το πρόβλημα των vanishing και exploding gradients. Η αρχιτεκτονική που χρησιμοποιήθηκε τελικά είναι μια συνηθισμένη δομή για LSTM όπου στην συγκεκριμένη περίπτωση αποτελείται από 4 επίπεδα LSTM με 100 νευρώνες το κάθε ένα, 3 dropout επίπεδα για να αποφευχθεί το overfitting κι ένα Dense επίπεδο με ένα νευρώνα. Η βελτιστοποίηση Adam είναι μια sgd (stochastic gradient descent) μέθοδος που βασίζεται στην προσαρμοστική εκτίμηση των ροπών πρώτης και δεύτερης τάξης. Ως σφάλμα για το training χρησιμοποιήθηκε το mean squared error.



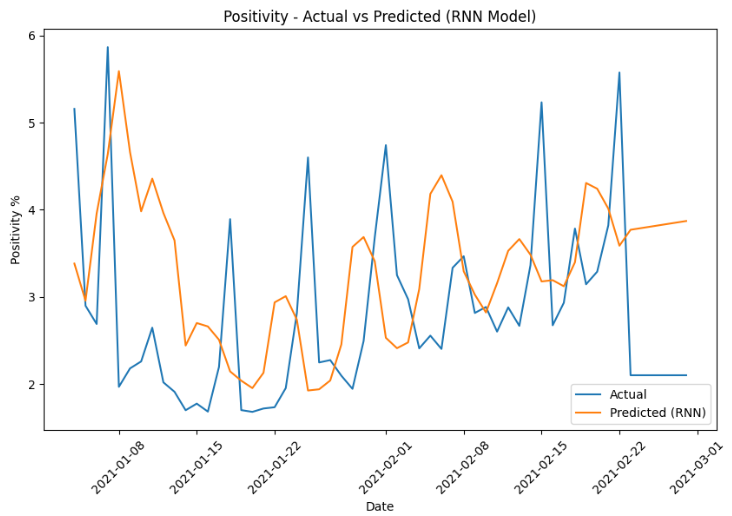
Για τον SVM παλινδρομητή χρησιμοποιήθηκε η κλάση SVR με τον πυρήνα rbf. Οι πυρήνες RBF τοποθετούν μια συνάρτηση ακτινικής βάσης με κέντρο κάθε σημείο και στη συνέχεια εκτελούν γραμμικούς χειρισμούς για να αντιστοιχίσουν τα σημεία σε χώρους υψηλότερων διαστάσεων που είναι ευκολότερο να διαχωριστούν.

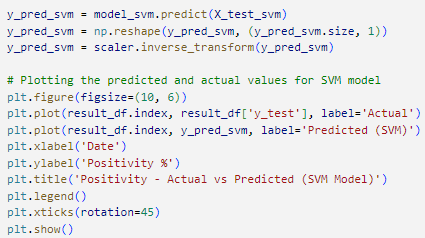


Σε αυτό το κομμάτι του κώδικα φτιάχνουμε ένα dataframe το οποίο μας βολεύει στη συνέχεια ώστε να κάνουμε γραφική παράσταση actual vs predicted values και στον x άξονα να έχουμε την ημερομηνία αντί για τον αριθμό των δειγμάτων. Κάνουμε και το απαραίτητο reshape και inverse transform ώστε να έχουμε τις σωστές τιμές.

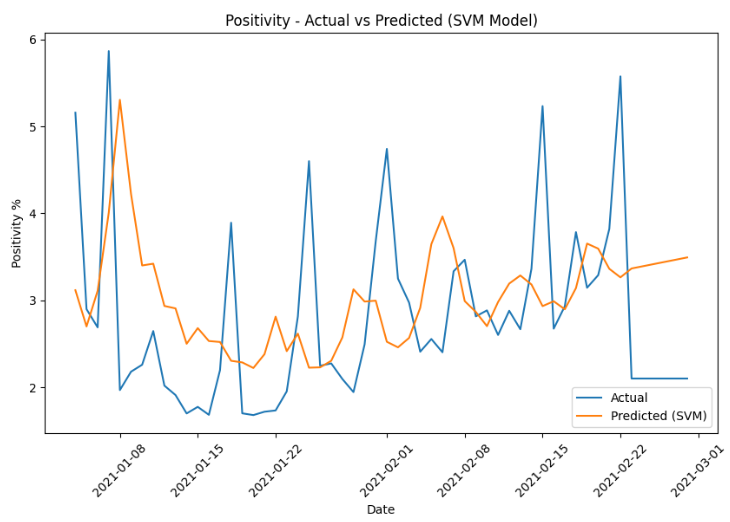


Με αυτό το κομμάτι κώδικα κάνουμε το plotting actual vs predicted positivity για το RNN.





Με αυτό το κομμάτι κώδικα κάνουμε το plotting actual vs predicted positivity για το SVM.



Από τις γραφικές παραστάσεις παρατηρούμε ότι και οι δύο παλινδρομητές μπορούν να πραγματοποιήσουν αρκετά ικανοποιητική καθημερινή ανάλυση για την πρόβλεψη της θετικότητας για 3 μέρες μετά.



Οι τιμές των μετρικών είναι αρκετά κοντά μεταξύ τους οπότε παρατηρούμε ότι το SVM υπερισχύει ελάχιστα. Επίσης παρατηρούμε ότι τα εκπαιδευμένα μοντέλα μας μπορούν να αποδώσουν και για νέα δεδομένα και να αποφέρουν καλά αποτελέσματα.